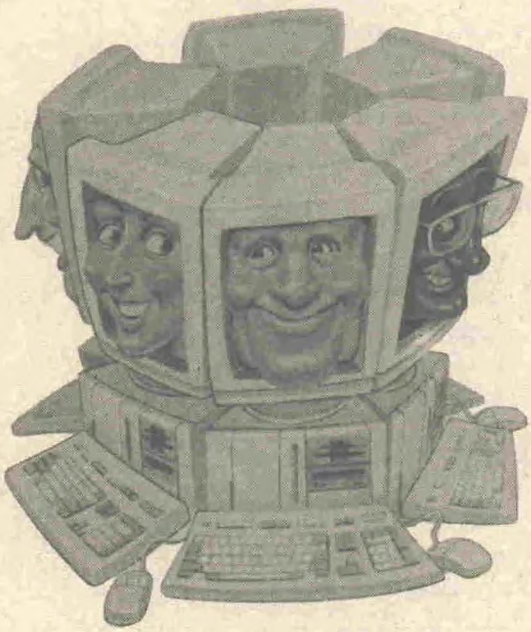
ธาวินทร์ ลีลาคุณารักษ์

การสร้างหน้าจอ (Form) ที่ใช้ตารางข้อมูลแบบ Master-Detail ใน Delphi



การสร้างหน้าจอที่มีการใช้ตารางข้อมูลในลักษณะแบบ Master-Detail โดยใช้ Delphi

ในการสร้างโปรแกรมคอมพิวเตอร์สำหรับใช้ในกิจการทั่วไปนั้น คงจะหนีไม่พ้นเรื่องของการจัดการฐานข้อมูล ซึ่งจะใช้รวบรวมข้อมูลต่าง ๆ ที่เกิดขึ้นในกิจการไว้เป็นระบบ เพื่อความสะดวกในการควบคุมและสามารถดึงมาใช้งานในลักษณะต่าง ๆ ได้ง่ายขึ้น

การสร้างโปรแกรมที่เกี่ยวข้องกับระบบฐานข้อมูลโดยทั่วไป จะมีงานที่เราพบเห็นอยู่บ่อย ๆ งานหนึ่ง ซึ่งก็คือ การสร้างหน้าจอ (Form) หรือส่วน User Interface ของโปรแกรมที่มีการใช้ตารางข้อมูล (Table) มากกว่าหนึ่งตาราง โดยตารางข้อมูลดังกล่าวจะมีการกำหนดความสัมพันธ์เป็นแบบ One-To-Many หรือที่เรียกกันทั่วไปว่า Master-Detail Tables

ตัวอย่างของงานที่มีการใช้ตารางข้อมูลแบบ Master-Detail ที่เชื่อว่าหลาย ๆ ท่านคงจะคุ้นเคยกันเป็นอย่างดี ได้แก่ โปรแกรมที่ใช้สำหรับจัดทำใบสั่งซื้อสินค้า (Purchase Order:PO) หรือที่เรียกกันสั้น ๆ ว่าโปรแกรมทำใบพีโอ โดยทั่วไปแล้วข้อมูลที่ปรากฏในใบสั่งซื้อสินค้านั้น จะมาจากตารางข้อมูลอย่างน้อย 2 ชุด คือ ชุดแรกได้แก่ ข้อมูลที่เป็นรายละเอียดส่วนหัวของใบสั่งซื้อสินค้า ซึ่งจะประกอบด้วย เลขที่เอกสาร วันที่ทำเอกสาร ชื่อที่อยู่บริษัทที่จะซื้อสินค้า เป็นต้น ส่วนข้อมูลชุดที่สองของใบสั่งซื้อสินค้า ได้แก่ รายละเอียดของสินค้าที่จะสั่งซื้อ ซึ่งจะประกอบด้วย รหัสสินค้า ชื่อสินค้า ราคา และจำนวนที่สั่งซื้อ เป็นต้น

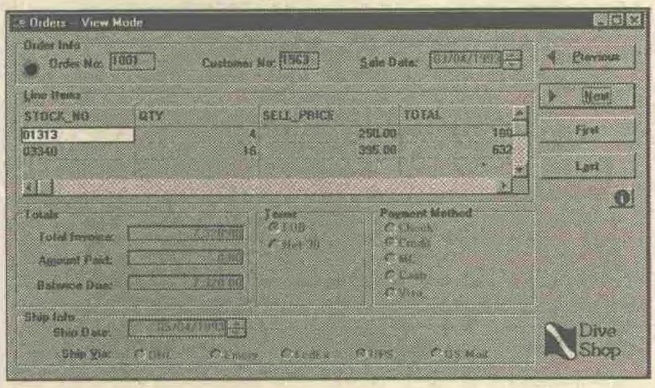
ลักษณะสำคัญของโปรแกรมที่มีการใช้ตารางข้อมูลแบบ Master-Detail ก็คือ เมื่อเราเลื่อนตัวชี้รายการ (Record Pointer) ในตารางส่วนที่เป็น Master ไปยังรายการถัดไปหรือไปยังรายการอื่น ๆ กลุ่มข้อมูลในตารางข้อมูลส่วน Detail ก็จะเปลี่ยนไปด้วย แต่ถ้าเราไปเลื่อนตัวชี้รายการของตารางข้อมูลส่วน Detail ข้อมูลในตารางข้อมูลส่วน Master จะไม่มีการเปลี่ยนแปลงหรือขยับไปรายการอื่น ๆ

และลักษณะที่สำคัญของ Master-Detail Tables อีกประการหนึ่ง คือ ถ้าเราสังเกตของมูลในตารางข้อมูลส่วน Detail จะพบว่า ข้อมูลในส่วนนี้จะเกี่ยวข้องกับรายการข้อมูลในตารางข้อมูลส่วน Master เฉพาะรายการใดรายการหนึ่งเท่านั้น จะไม่มีการปนกัน ในรูปที่ 2 จะเป็นตัวอย่างของข้อมูล ที่จัดเก็บในตารางข้อมูลที่มีลักษณะ Master-Detail ตามที่กล่าวมาในข้างต้น

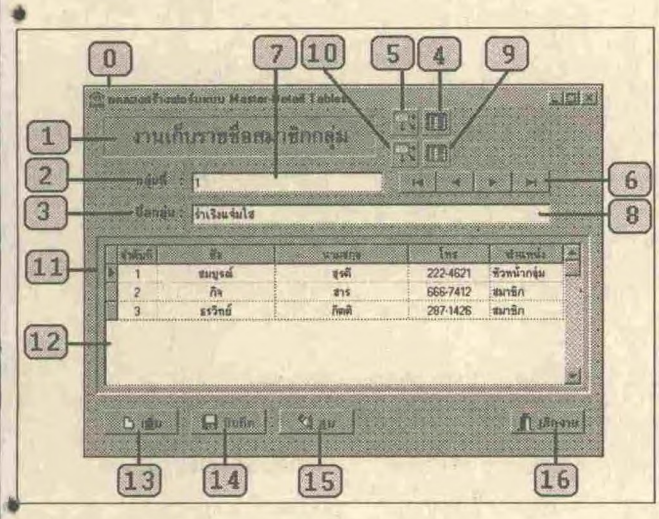
การทำงานบน Delphi

ในการทำหน้าจอของโปรแกรม โดยเฉพาะส่วนที่มีการใช้ตารางข้อมูลแบบ Master-Detail นี้ ไม่ใช่เรื่องง่ายในการสร้างโปรแกรมในยุคก่อน ๆ ถ้าเราสร้างโปรแกรมด้วย Clipper หรือ Foxpro ก็จะง่ายหน่อย เพราะภาษาโปรแกรมทั้งสองชุดนี้ จะมีฟังก์ชัน DBEdit() หรือ

Browse() ที่สามารถสร้างชุดคำสั่งแทรกลงไปได้ให้ใช้ แต่ถ้าเป็นการสร้างโปรแกรมด้วยภาษา Pascal หรือ C เราจะต้องเป็นคนเขียนโปรแกรมสร้างส่วนประกอบต่าง ๆ เองทั้งหมด



รูปที่ 1 ตัวอย่างแอพพลิเคชันที่ใช้ตารางข้อมูลแบบ Master-Detail



รูปที่ 3 โปรแกรมเก็บข้อมูลกลุ่มกิจกรรม

มาถึงยุคของการสร้างโปรแกรมที่มีการใช้เทคนิค Visual Programming อย่างในปัจจุบัน งานสร้างหน้าจอที่กล่าวมาในข้างต้นดูจะง่ายกว่า เพียงเราจัดทำการจัดองค์ประกอบต่าง ๆ ที่คอมไพเลอร์เตรียมไว้ให้อย่างเหมาะสม ส่วน User Interface หรือหน้าจอของโปรแกรมที่มีการเชื่อมโยงกับตารางข้อมูลในลักษณะ Master-Detail ก็จะเสร็จสมบูรณ์ได้อย่างไม่ยากเย็น

ตัวอย่าง

ผมมีตัวอย่างง่าย ๆ พอให้เป็นแนวทางในการนำไปประยุกต์ใช้กับงานอื่น ๆ ได้ โดยในตัวอย่างชุดนี้ จะเป็นโปรแกรมสำหรับจัดเก็บข้อมูลเกี่ยวกับกลุ่มกิจกรรม ที่กำหนดขึ้นเพื่อทำกิจกรรมใดกิจกรรมหนึ่ง โดยภายในโปรแกรมจะมีการเก็บลำดับที่ของกลุ่ม (GroupNo) และชื่อของโปรแกรม (GroupName) ไว้ในตารางข้อมูลส่วน Master โดยในที่นี้จะใช้ลำดับที่ของกลุ่มเป็นคีย์ (Key) หลักของงาน ส่วนตารางข้อมูลในส่วน Detail จะมีการเก็บรายชื่อของสมาชิกแต่ละคนภายในกลุ่ม ซึ่งหน้าตาของโปรแกรมและชิ้นส่วนต่าง ๆ ที่จะนำมาประกอบเป็นตัวโปรแกรมชุดนี้ จะอยู่ในรูปที่ 3

โครงสร้างตารางข้อมูล

ในโปรแกรมชุดนี้จะมีการใช้ตารางข้อมูลจำนวน 2 ชุด ในลักษณะของ Master-Detail ตามที่กล่าวมาในข้างต้น ตารางข้อมูลส่วน Master จะจัดเก็บในแฟ้มชื่อ MASTER.DB ส่วนตารางข้อมูลในส่วน Detail จะจัดเก็บในแฟ้มชื่อ DETAIL.DB โดยตารางข้อมูลทั้งสองชุดนี้จะเป็นตารางข้อมูลแบบ Paradox ซึ่งรายละเอียดโครงสร้างภายในของตารางข้อมูลแต่ละชุดมีดังนี้ (สร้างในโปรแกรม Database Desktop ที่มากับ Delphi)

MASTER TABLE

CODE	NAME	TPRICE
10101	ชุดของขวัญชุดที่1	247.00
10102	ชุดของขวัญชุดที่2	300.00
10103	ชุดของขวัญชุดที่3	2,000.00
...	...	...
...	...	...

DETAIL TABLE

CODE	DETAIL CODE	NAME	PRICE
10101	74012	กาแฟเขาช่อง	85.00
10101	74015	เครื่องดื่ม โอวัลติน	102.00
10101	74022	นม UHT ไทยเดนมาร์ก	60.00
10102	32441	องุ่นแดงไม่มีเมล็ด	200.00
10102	32571	แอปเปิลฟูจิ80	40.00
10102	32745	กีวีฟลุต	60.00
10103	65447	กระติกน้ำไฟฟ้า	800.00
10103	65777	เครื่องปิ้งขนมปัง	400.00
10103	...	...	...
...	...	...	...

รูปที่ 2 ตัวอย่างข้อมูลที่เก็บในลักษณะ Master-Detail Tables

ตัวอย่างที่ 1

```
MASTER.DB

Field Name Type Size Key

1 GroupNo A 4 *
2 GroupName A 30

**********

DETAIL.DB

Field Name Type Size Key

1 GroupNo A 4 *
2 Order N *
3 Name A 30
4 LastName A 30
5 Phone A 9
6 Position A 20

**********
```

ข้อสังเกตในการสร้างตารางข้อมูลที่จะนำไปใช้งานในลักษณะ Master-Detail ที่น่าสนใจ คือ ในการสร้างฟิลด์ที่ใช้เป็นคีย์ของตารางข้อมูลในส่วน Detail นั้น จะต้องประกอบด้วยฟิลด์ที่ใช้เป็นคีย์อย่างน้อยสองชุด โดยที่ฟิลด์ข้อมูลที่เป็นคีย์ชุดแรก จะต้องมีรายละเอียดภายในฟิลด์เหมือนกับฟิลด์ที่ใช้เป็นคีย์ของตารางข้อมูลส่วน Master ซึ่งคีย์ชุดแรกนี้จะใช้สำหรับเชื่อมโยงข้อมูลทั้งสองตารางเข้าด้วยกัน (One-To-Manay Relationship) ส่วนฟิลด์ที่ใช้เป็นคีย์ชุดที่สองของตารางข้อมูลส่วน Detail จะเป็นคีย์ที่ใช้ภายในตารางข้อมูลชุดที่สองเอง ดังรายละเอียดโครงสร้างของตารางข้อมูลที่ปรากฏในข้างต้น

การปรับแต่งคุณสมบัติชิ้นส่วนต่าง ๆ

สำหรับการปรับแต่งค่าคุณสมบัติ (Properties) ของชิ้นส่วนโปรแกรมต่าง ๆ ที่เรานำมาประกอบเป็นหน้าตาของโปรแกรม ดังที่ปรากฏในรูปที่ 3 จะมีรายละเอียดดังตัวอย่างที่ 2

ตัวอย่างที่ 2

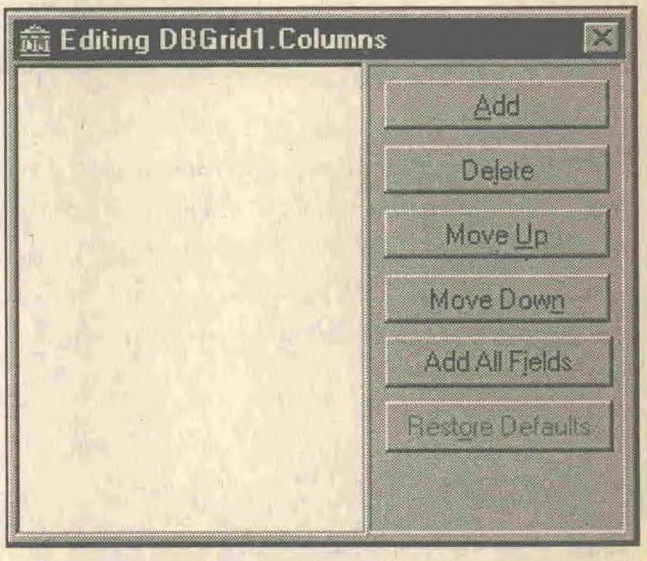
```
ชิ้นส่วนแบบ จากต้นแบบชุด ชื่อคุณสมบัติ ค่าที่กำหนดใหม่
 (Component)  (Pallate)  (Properties) (NewValue)

 0)  Form - Caption ทดลองสร้างฟอร์มแบบ Master-Detail Tables

 1)  Panel Standard BevelOuter bvLowered
          Caption งานเก็บรายชื่อสมาชิกกลุ่ม
          Font Font : EucrosiaUPC
          Font style : Bold
          Size : 22
          Color : Blue

 2)  Label Standard Caption กลุ่มที่ :
          Font Font : System
          Font Style : Bold
          Font Size : 8
          Color : Blue

 3)  Label Standard Caption ชื่อกลุ่ม :
          Font Font : System
          Font Style : Bold                    →
```

ตัวอย่างที่ 2 (ต่อ)

```
          Font Size : 8
          Color : Blue

 4)  Table[Table1] Data Access DatabaseName c:
          TableName master.db
          Active True

 5)  DataSource [1] Data Access DataSet Table1

 6)  DBNavigator Data Controls DataSource
DataSource1
          ShowHint True
          VisibleButtons nbInsert : False
          nbDelete : False
          nbEdit : False
          nbPost : False
          nbCancel: False
          nbRefresh : False

 7)  DBEdit Data Controls DataSource DataSource1
          DataField GroupNo

 8)  DBEdit Data Controls DataSource DataSource1
          DataField GroupName

 9)  Table [Table2] Data Access DatabaseName c:
          TableName detail.db
          MasterSource DataSource1
          MasterFields GroupNo
          IndexFieldNames GroupNo
          Active True

10)  DataSource [2] Data Access DataSet Table2

11)  Panel Standard Caption ""

12)  DBGrid1 Data Controls DataSource DataSource2
          Columns [ให้กำหนดในกรอบ Edit Columns]

13)  BitBtn Additional Name AddBitBtn
          Caption เ&พิ่ม
          Font Font : System
          Font style : Bold
          Size : 10
          Color : Blue
          Glyph [เลือกแฟ้ม Filenew.bmp จาก
                ..\Images\Buttons ของ
                Delphi]

14)  BitBtn Additional Name SaveBitBtn
          Caption &บันทึก
          Font Font : System
          Font style : Bold
          Size : 10
          Color : Green
          Glyph [เลือกแฟ้ม Floppy.bmp จาก
                ..\Images\Buttons ของ
                Delphi]

15)  BitBtn Additional Name DelBitBtn
          Caption &ลบ
          Font Font : System
          Font style : Bold
          Size : 10
          Color : Red
          Glyph [เลือกแฟ้ม Erase.bmp จาก
                ..\Images\Buttons ของ
                Delphi]

16)  BitBtn Additional Kind bkClose
          Caption &เลิกงาน
          Font Font : System
          Font style : Bold
          Size : 10
          Color : Blue                         ■
```

ในการปรับแต่งค่าของคุณสมบัติของชิ้นส่วนต่าง ๆ ควรจะดำเนินการตามลำดับที่ปรากฏในตัวอย่างข้างต้น เนื่องจากค่าคุณสมบัติของชิ้นส่วนโปรแกรมในลำดับต้น ๆ จะมีผลกับชิ้นส่วนอื่น ๆ ที่อยู่ในลำดับถัดมา จากรายละเอียด การกำหนดค่าคุณสมบัติของชิ้นส่วนในข้างต้น ขอให้สังเกตดูความแตกต่างในการกำหนดค่าคุณสมบัติให้กับชิ้นส่วนโปรแกรมแบบ Table ที่ใช้เชื่อมโยงกับตารางข้อมูล โดยชิ้นส่วนโปรแกรมแบบ Table ที่ใช้เชื่อมโยงกับแฟ้ม MASTER.DB นั้น จะไม่มีการกำหนดค่าให้กับค่าคุณสมบัติในหัวข้อ MasterSource และ MasterField ไว้ แต่ในชิ้นส่วนโปรแกรมแบบ Table ที่ใช้เชื่อมโยงกับแฟ้ม DETAIL.DB (Table2) จะมีการกำหนดค่าของคุณสมบัติทั้งสองหัวข้อไว้ ซึ่งการกำหนดค่าคุณสมบัติในลักษณะนี้ จะเป็นการกำหนดความสัมพันธ์แบบ One-To-Many ให้กับตารางข้อมูลทั้งสองชุดโดยอัตโนมัติ

สำหรับการปรับแต่งค่าคุณสมบัติของชิ้นส่วนแบบ DBGrid ที่ใช้แสดงและรับค่าจากตารางข้อมูลส่วน Detail ไม่ว่าจะเป็นการกำหนดข้อความที่แสดงในส่วนหัวตารางของ DBGrid หรือการจัดตำแหน่งของข้อมูลที่จะปรากฏในแต่ละช่อง งานทั้งหมดนี้จะกระทำได้โดยใช้เมาส์ดับเบิลคลิกที่ตัว DBGrid ซึ่งจะมีกรอบย่อย ๆ สำหรับกำหนดคอลัมน์ให้กับ DBGrid ขึ้น ให้คลิกที่ปุ่มคำสั่ง Add All Fields ของกรอบกำหนดคอลัมน์ดังกล่าว จากนั้นให้ลบชื่อฟิลด์ที่ใช้เชื่อมโยงกับตารางข้อมูลส่วน Master ออก ซึ่งในที่นี้คือ ฟิลด์ GroupNo เนื่องจากเราไม่ต้องการให้ปรากฏในส่วน DBGrid เพราะเวลาที่ผู้ใช้ทำงานจะเกิดความสับสนและเกิดข้อผิดพลาดได้ง่าย ส่วนการกำหนดค่าคุณสมบัติของแต่ละคอลัมน์สามารถใช้เมาส์คลิกที่ชื่อฟิลด์ที่ต้องการในกรอบข้างต้น แล้วไปปรับแต่งค่าต่าง ๆ ใน Object Inspecter ของ Delphi ตามความเหมาะสมได้ทันที



รูปที่ 4 กรอบ Edit Columns ใน DBGrid ของ Delphi

การสร้างชุดคำสั่งควบคุมโปรแกรม

สำหรับจุดที่ต้องสร้างชุดคำสั่งแทรกลงไป เพื่อควบคุมกลไกการทำงานของชิ้นส่วนต่าง ๆ ของโปรแกรมชุดนี้ จะมีเพียง 3 จุดเท่านั้น คือ ที่ปุ่มคำสั่ง "เพิ่ม" ปุ่มคำสั่ง "บันทึก" และที่ปุ่มคำสั่ง "ลบ" ตามลำดับ ส่วนที่ปุ่มคำสั่ง "เลิกงาน" นั้น ไม่ต้องแทรกชุดคำสั่งใด ๆ ลงไป เพราะเราได้เลือกใช้กลไกการทำงานมาตรฐานของ Delphi ไว้แล้ว โดยการกำหนดค่าคุณสมบัติในหัวข้อ "Kind" ให้เป็น "kbClose" ดังที่ปรากฏในรายการการปรับแต่งคุณสมบัติข้างต้น

การทำงานที่ปุ่มคำสั่ง "เพิ่ม"

งานที่เราต้องการให้เกิดขึ้นหลังจากที่ผู้ใช้คลิกที่ปุ่มคำสั่งชุดนี้คือ โปรแกรมจะไปสร้างรายการว่าง (New Record) ชุดใหม่ขึ้นให้กับตารางข้อมูลส่วน Master โดยรายการว่างดังกล่าวจะไปอยู่ตอนท้ายสุดของตารางข้อมูล ชุดคำสั่งที่จะแทรกที่จุดนี้มีดังนี้ (เฉพาะส่วนตัวหนา)

```
procedure TForm1.AddBitBtnClick (Sender: TObject);
begin
  Table1.Append;
  DBedit1.SetFocus;
end;
```

การเพิ่มรายการข้อมูลลงในตารางข้อมูลในส่วน Detail ของโปรแกรม เราไม่จำเป็นต้องสร้างชุดคำสั่งใด ๆ ไว้ เพราะโปรแกรมจะทำการเตรียมพื้นที่ว่างไว้ให้อัตโนมัติ โดยโปรแกรมจะสร้างรายการว่างดังกล่าวทันทีที่เคอร์เซอร์ของโปรแกรมอยู่ที่รายการสุดท้ายของตารางข้อมูล แล้วผู้ใช้กดปุ่มลูกศรลง หรือคลิกที่รายการด้านล่างสุดของ DBGrid และเมื่อเราใส่ข้อมูลครบทุกคอลัมน์แล้ว หากมีการเลื่อนตำแหน่งตัวชี้รายการของตารางข้อมูล โปรแกรมจะทำการจัดเก็บข้อมูลที่ใส่ลงไปใหม่ให้อัตโนมัติ

ข้อกำหนดอย่างหนึ่งในการเพิ่มรายการลงในตารางข้อมูลส่วน Detail คือ ทุกครั้งที่มีการเพิ่มรายการใหม่ลงไป เราจะต้องนำค่าภายในฟิลด์ที่เป็นคีย์ของตารางข้อมูลส่วน Master มาใส่ในฟิลด์ที่ใช้เชื่อมโยงความสัมพันธ์ในตารางข้อมูลส่วน Detail ด้วย (คีย์ชุดแรก) ซึ่งเป็นเรื่องโชคดีอย่างหนึ่ง ที่การสร้างโปรแกรมใน Delphi เราไม่ต้องสร้างชุดคำสั่งเพื่อทำงานนี้เอง เพราะ Delphi จะทำงานนี้ให้อัตโนมัติ แต่ถ้าเราสร้างโปรแกรมด้วย Visual BASIC เราจะต้องสร้างชุดคำสั่งเพื่อทำงานข้างต้นขึ้นเอง

การทำงานที่ปุ่มคำสั่ง "บันทึก"

ปุ่มคำสั่งบันทึกชุดนี้ จะทำหน้าที่จัดเก็บข้อมูลที่ค้างอยู่ในหน่วยความจำของเครื่องคอมพิวเตอร์ลงในฮาร์ดดิสก์ โดยปกติโปรแกรมจะทำการบันทึกข้อมูลให้อัตโนมัติ ทันทีที่มีการเลื่อนตำแหน่งตัวชี้รายการ (Record Pointer) ของตารางข้อมูล แต่เพื่อความแน่ใจ เราควรจะสร้างปุ่มคำสั่งบันทึกนี้ไว้ เผื่อในกรณีที่ผู้ใช้ใส่ข้อมูลต่าง ๆ ไว้ แล้วไม่มีการเลื่อนตำแหน่งตัวชี้รายการ แต่ปิดโปรแกรมไปก่อน ซึ่งกรณีนี้ข้อมูลที่ผู้ใช้ใส่ไว้หลังสุดจะหายไปทันที ชุดคำสั่งที่จะแทรกที่จุดนี้มีดังนี้

```
procedure TForm1.SaveBitBtnClick (Sender:
TObject) ;
begin
  if  (Table1.State=dsEdit) or
(Table1.State=dsInsert)  then
   Table1.Post;
  if  (Table2.State=dsEdit) or
(Table2.State=dsInsert)  then
   Table2.Post;
end;
```

ในการใช้คำสั่ง Post เพื่อบันทึกข้อมูลที่อยู่หน่วยความจำลงฮาร์ดดิสก์นั้น สถานะของตารางข้อมูล (Table State) ที่จะใช้คำสั่งนี้จะต้องอยู่ในสถานะ "Edit" หรือ "Insert" เท่านั้น ถ้าหากเป็นสถานะอื่น ๆ โปรแกรมจะเกิดข้อผิดพลาดขึ้นทันที ดังนั้นก่อนใช้คำสั่ง Post กับตารางข้อมูลใด ๆ เราจำเป็นต้องตรวจสอบก่อนว่าตารางข้อมูลชุดนั้น ๆ พร้อมที่จะทำการ Post หรือไม่ โดยตรวจสอบจากคุณสมบัติหัวข้อ State ของตารางข้อมูล ส่วนตัวแปร dsEdit และ dsInsert ที่เห็นในข้างต้น เป็นชื่อของค่าคงที่ที่ตรงกับสถานะ "Edit" หรือ "Insert" ของตารางข้อมูล ซึ่ง Delphi จัดเตรียมไว้ให้ เราสามารถนำมาใช้งานได้ทันที

การทำงานที่ปุ่มคำสั่ง "ลบ"

สำหรับปุ่มคำสั่ง "ลบ" จะทำหน้าที่ลบรายการข้อมูลที่ตัวชี้รายการชี้อยู่ออกจากตารางข้อมูล โดยในที่นี้จะดูที่ตารางข้อมูลส่วน Master เป็นหลัก ซึ่งในการลบข้อมูลออกจากตารางข้อมูลส่วน Master นั้น เราจะต้องไปลบข้อมูลที่เกี่ยวข้องในตารางข้อมูลส่วน Detail ออกด้วย ส่วนกรณีที่ผู้ใช้ต้องการลบรายการที่อยู่ในตารางข้อมูลส่วน Detail ที่แสดงอยู่ใน DBGrid จะสามารถทำได้โดยใช้เมาส์คลิกที่รายการที่จะทำการลบใน DBGrid แล้วกด Ctrl + Delete ตามลำดับ ดังนั้นในตัวอย่างชุดนี้ จึงไม่ได้สร้างชุดคำสั่งเพื่อทำงานดังกล่าวไว้ สำหรับชุดคำสั่งที่จะแทรกที่ปุ่มคำสั่ง "ลบ" ของโปรแกรมชุดนี้ มีดังนี้

```
procedure TForm1.DelBitBtnClick (Sender:TObject);
begin
  while  (Table2['GroupNo']=DBEdit1.Text)  do
   Table2.Delete;
  Table1.Delete;
end;
```

การอ้างอิงค่าภายในฟิลด์ของตารางข้อมูล เราสามารถอ้างโดยใส่ชื่อฟิลด์ไว้ใน "[..]" แล้วนำมาต่อกับชื่อตารางข้อมูลที่จะอ้างอิงได้ทันที ตามลักษณะที่ปรากฏในข้างต้น การอ้างอิงลักษณะนี้จะเรียกว่า Field Value ซึ่งเป็นการนำค่าภายในฟิลด์ข้อมูลมาใช้งานที่สะดวกมากวิธีหนึ่ง

จบบริบูรณ์

ทั้งหมดก็เป็นตัวอย่างง่าย ๆ ของโปรแกรมสร้างโปรแกรมที่มีการใช้ตารางข้อมูลในลักษณะ Master-Detail หรือตารางข้อมูลที่มีกำหนดความสัมพันธ์แบบ One-To-Many ซึ่งน่าจะเป็นประโยชน์กับการพัฒนางานของท่านตามสมควร พบกันใหม่โอกาสหน้าครับ